

CRAY
THE SUPERCOMPUTER COMPANY

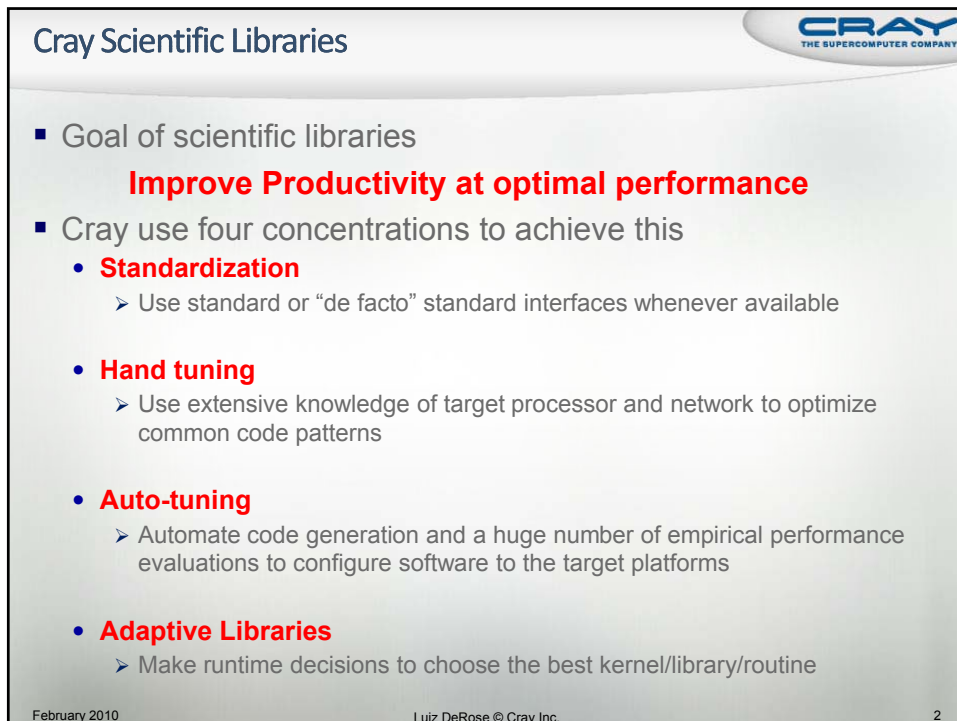
Cray Scientific Libraries

Luiz DeRose
Programming Environments Director
Cray Inc.

Joint Cray XT5 Workshop
NERSC / ORNL / NICS

Luiz DeRose © Cray Inc.

February, 2010



CRAY
THE SUPERCOMPUTER COMPANY

Cray Scientific Libraries

- Goal of scientific libraries
 - Improve Productivity at optimal performance**
- Cray use four concentrations to achieve this
 - **Standardization**
 - Use standard or “de facto” standard interfaces whenever available
 - **Hand tuning**
 - Use extensive knowledge of target processor and network to optimize common code patterns
 - **Auto-tuning**
 - Automate code generation and a huge number of empirical performance evaluations to configure software to the target platforms
 - **Adaptive Libraries**
 - Make runtime decisions to choose the best kernel/library/routine

February 2010

Luiz DeRose © Cray Inc.

2

Standardization



- Three separate classes of standardization, each with a corresponding definition of productivity
 1. Standard interfaces (e.g., dense linear algebra)
 - Bend over backwards to keep everything the same despite increases in machine complexity. Innovate 'behind-the-scenes'
 - Productivity -> innovation to keep things simple
 2. Adoption of near-standard interfaces (e.g., sparse kernels)
 - Assume near-standards and promote those. Out-mode alternatives. Innovate 'behind-the-scenes'
 - Productivity -> innovation in the simplest areas
 - (requires the same innovation as #1 also)
 3. Simplification of non-standard interfaces (e.g., FFT)
 - Productivity -> innovation to make things simpler than they are

February 2010

Luiz DeRose © Cray Inc.

3

Hand Tuning

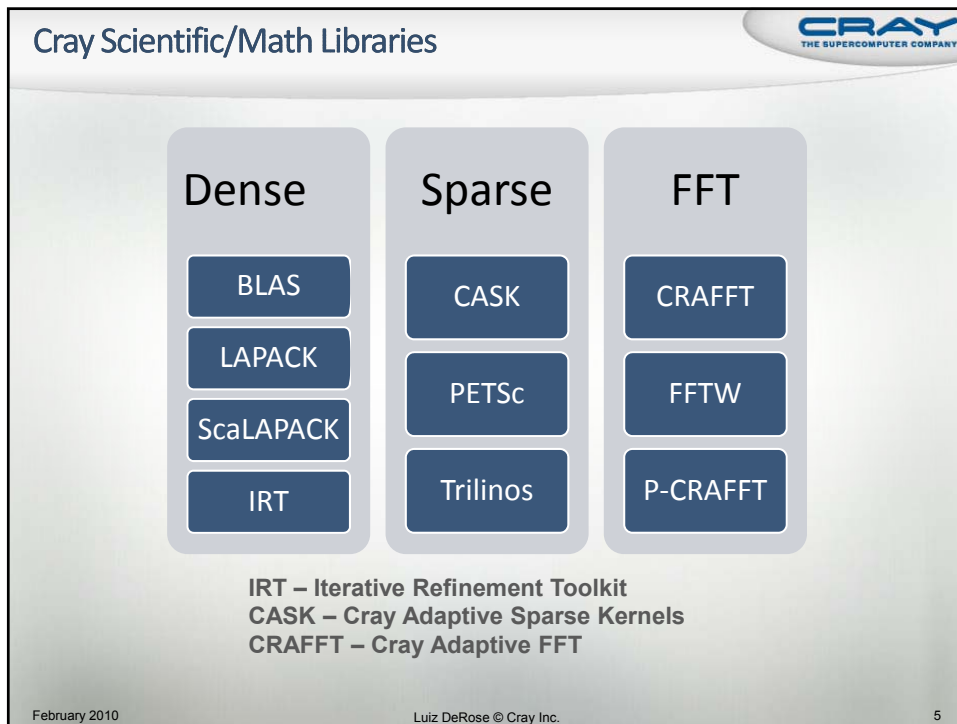


- Algorithmic tuning
 - Increased performance by exploiting algorithmic improvements
 - Sub-blocking, new algorithms
 - **LAPACK, ScaLAPACK**
- Kernel tuning
 - Improve the numerical kernel performance in assembly language
 - **BLAS, FFT**
- Parallel tuning
 - Exploit Cray's custom network interfaces and MPT
 - **ScaLAPACK, P-CRAFFT**

February 2010

Luiz DeRose © Cray Inc.

4



PETSc (Portable, Extensible Toolkit for Scientific Computation)

The slide features the PETSc logo and a list of its capabilities:

- Serial and Parallel versions of sparse iterative linear solvers
 - Suites of iterative solvers
 - CG, GMRES, BiCG, QMR, etc.
 - Suites of preconditioning methods
 - IC, ILU, diagonal block (ILU/IC), Additive Schwartz, Jacobi, SOR
 - Support block sparse matrix data format for better performance
 - Interface to external packages (ScaLAPACK, SuperLU_DIST)
 - Fortran and C support
 - Newton-type nonlinear solvers
- Large user community
 - DoE Labs, PSC, CSCS, CSC, ERDC, AWE and more.
- <http://www-unix.mcs.anl.gov/petsc/petsc-as>

February 2010 Luiz DeRose © Cray Inc. 6

Usage and External Packages



- Cray provides state-of-the art scientific computing packages to strengthen the capability of PETSc
 - Hypre: scalable parallel preconditioners
 - AMG (Very scalable and efficient for specific class of problems)
 - 2 different ILU (General purpose)
 - Sparse Approximate Inverse (General purpose)
 - ParMetis: parallel graph partitioning package
 - MUMPS: parallel multifrontal sparse direct solver
 - SuperLU: sequential version of SuperLU_DIST
- To use Cray-PETSc, load the appropriate module :
 - module load petsc**
 - module load petsc-complex**
 - (no need to load a compiler specific module)
- Treat the Cray distribution as your local PETSc installation

February 2010

Luiz DeRose © Cray Inc.

7

Trilinos



- The Trilinos Project <http://trilinos.sandia.gov/>
 - “an effort to develop algorithms and enabling technologies within an object-oriented software framework for the solution of large-scale, complex multi-physics engineering and scientific problems”
- A unique design feature of Trilinos is its focus on packages
- Very large user-base and growing rapidly
 - Important to DOE
- Cray's optimized Trilinos released on January 21
 - Includes 50+ Trilinos packages
 - Optimized via CASK
 - Any code that uses Epetra objects can access the optimizations
- Usage :
 - module load trilinos**

February 2010

Luiz DeRose © Cray Inc.

8

Cray Adaptive Sparse Kernel (CASK)



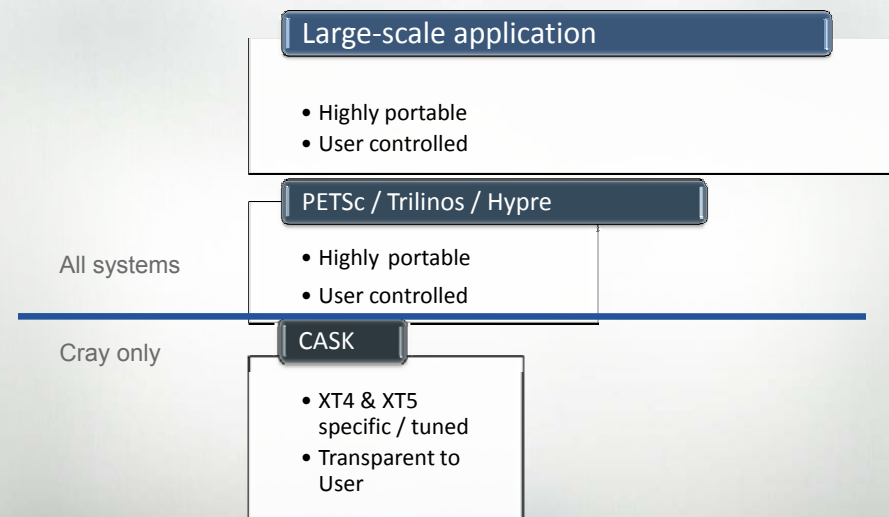
- CASK is a product developed at Cray using the Cray Auto-tuning Framework (Cray ATF)
- The CASK Concept :
 - Analyze matrix at minimal cost
 - Categorize matrix against internal classes
 - Based on offline experience, find best CASK code for particular matrix
 - Previously assign “best” compiler flags to CASK code
 - Assign best CASK kernel and perform Ax
- CASK silently sits beneath PETSc and Trilinos on Cray systems
- Released with PETSc 3.0 in February 2009
 - Generic and blocked CSR format

February 2010

Luiz DeRose © Cray Inc.

9

Support Model



February 2010

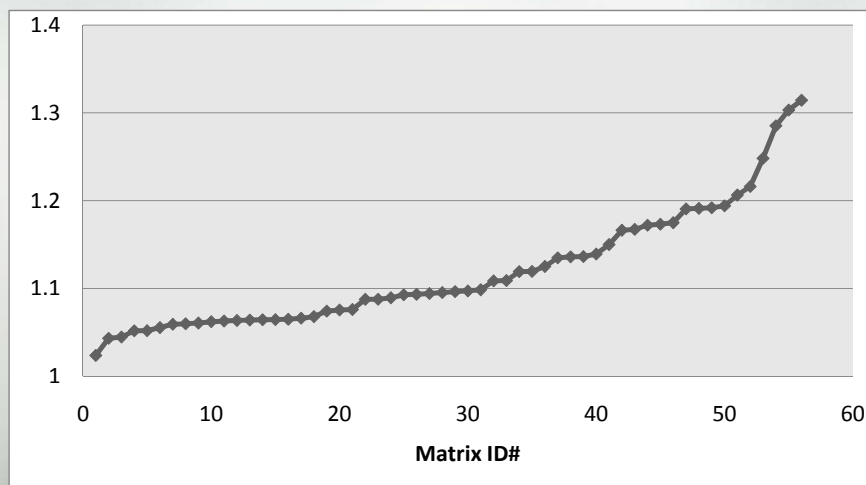
Luiz DeRose © Cray Inc.

10

CASK and PETSc: Single Node XT5



Speedup on Parallel SpMV on 8 cores, 60 different matrices



February 2010

Luiz DeRose © Cray Inc.

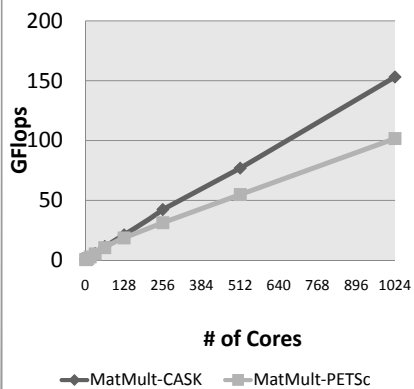
11

CASK + PETSc Scalability (XT4)



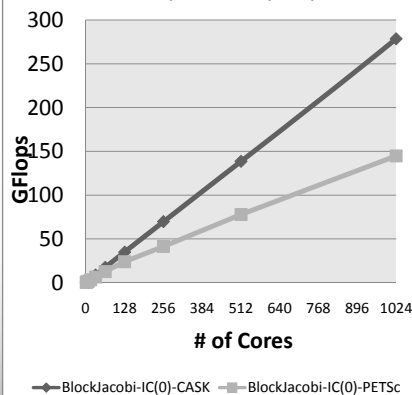
SpMV

Performance of CASK VS PETSc
N=65,536 to 67,108,864



Block Jacobi Preconditioning

Performance of CASK VS PETSc
N=65,536 to 67,108,864



February 2010

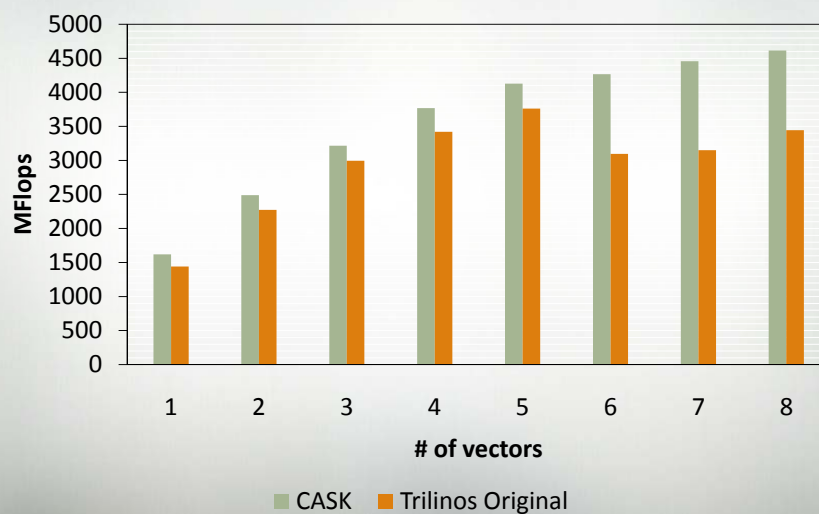
Luiz DeRose © Cray Inc.

12

Trilinos + CASK Multiple Vector SpMV Kernel Performance on single Istanbul CPU



Geometric Mean of 80 sparse matrix instances from U of Florida collection

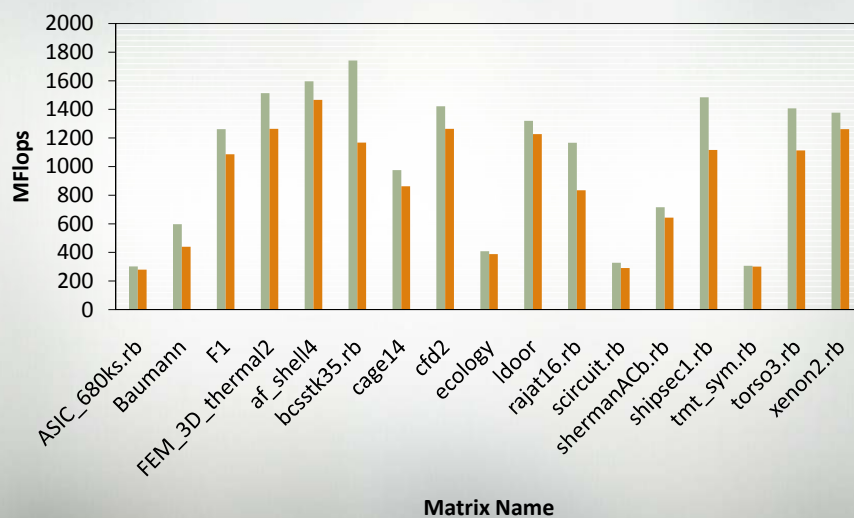


February 2010

Luiz DeRose © Cray Inc.

13

Trilinos +CASK SpMV performance on single Istanbul CPU



February 2010

Luiz DeRose © Cray Inc.

14

Cray Adaptive FFT (CRAFFT)



- In FFTs, the problems are
 - Which library choice to use?
 - How to use complicated interfaces (e.g., FFTW)
- Standard FFT practice
 - Do a plan stage
 - Deduced machine and system information and run micro-kernels
 - Select best FFT strategy
 - Do an execute

Our system knowledge can remove some of this cost!

February 2010

Luiz DeRose © Cray Inc.

15

CRAFFT library



- CRAFFT is designed with simple-to-use interfaces
 - Planning and execution stage can be combined into one function call
 - Underneath the interfaces, CRAFFT calls the appropriate FFT kernel
- CRAFFT provides both offline and online tuning
 - Offline tuning
 - Which FFT kernel to use
 - Pre-computed PLANS for common-sized FFT
 - No expensive plan stages
 - Online tuning is performed as necessary at runtime as well
- At runtime, CRAFFT will **adaptively select the best** FFT kernel to use based on both offline and online testing (e.g. FFTW, Custom FFT)

February 2010

Luiz DeRose © Cray Inc.

16

Performance of one CRAFFT feature - 3-d FFT times using FFTW wisdom under-the-covers



	128x128	256x256	512x512
FFTW plan	74	312	2758
FFTW exec	0.105	0.97	9.7
CRAFFT plan	0.00037	0.0009	0.00005
CRAFFT exec	0.139	1.2	11.4

Luiz DeRose © Cray Inc.

February 2010

Slide
17

CRAFFT usage



1. Load module fftw/3.2.0 or higher.
2. Add a Fortran statement "use crafft"
3. call crafft_init()
4. Call crafft transform using none, some or all optional arguments (as shown in red)

In-place, implicit memory management :

```
call crafft_z2z3d(n1,n2,n3,input,ld_in,ld_in2,sign)
```

in-place, explicit memory management

```
call crafft_z2z3d(n1,n2,n3,input,ld_in,ld_in2,sign,work)
```

out-of-place, explicit memory management :

```
crafft_z2z3d(n1,n2,n3,input,ld_in,ld_in2,output,ld_out,ld_out2,sign,work)
```

Note : the user can also control the planning strategy of CRAFFT using the CRAFFT_PLANNING environment variable and the do_exe optional argument, please see the intro_crafft man page.

February 2010

Luiz DeRose © Cray Inc.

18

Parallel CRAFFT



- As of December 2009, CRAFFT includes distributed parallel transforms
- Uses the CRAFFT interface prefixed by “p”, with optional arguments
- Can provide performance improvement over FFTW 2.1.5
- Currently implemented
 - complex-complex
 - 3-d and 2-d
 - In-place and out-of-place
- Next release of CRAFFT
 - Parallel real-complex and complex-real
 - C language support for serial and parallel

February 2010

Luiz DeRose © Cray Inc.

19

parallel CRAFFT usage



1. Add “use crafft” to Fortran code
2. Initialize CRAFFT using `crafft_init`
3. Assume MPI initialized and data distributed (see manpage)
4. Call `crafft`, e.g. (optional arguments in red)

2-d complex-complex, in-place, internal mem management :

```
call crafft_pz2z2d(n1,n2,input,sign,flag,comm)
```

2-d complex-complex, in-place with no internal memory :

```
call crafft_pz2z2d(n1,n2,input,sign,flag,comm,work)
```

2-d complex-complex, out-of-place, internal mem manager :

```
call crafft_pz2z2d(n1,n2,input,output,sign,flag,comm)
```

2-d complex-complex, out-of-place, no internal memory :

```
crafft_pz2z2d(n1,n2,input,output,sign,flag,comm,work)
```

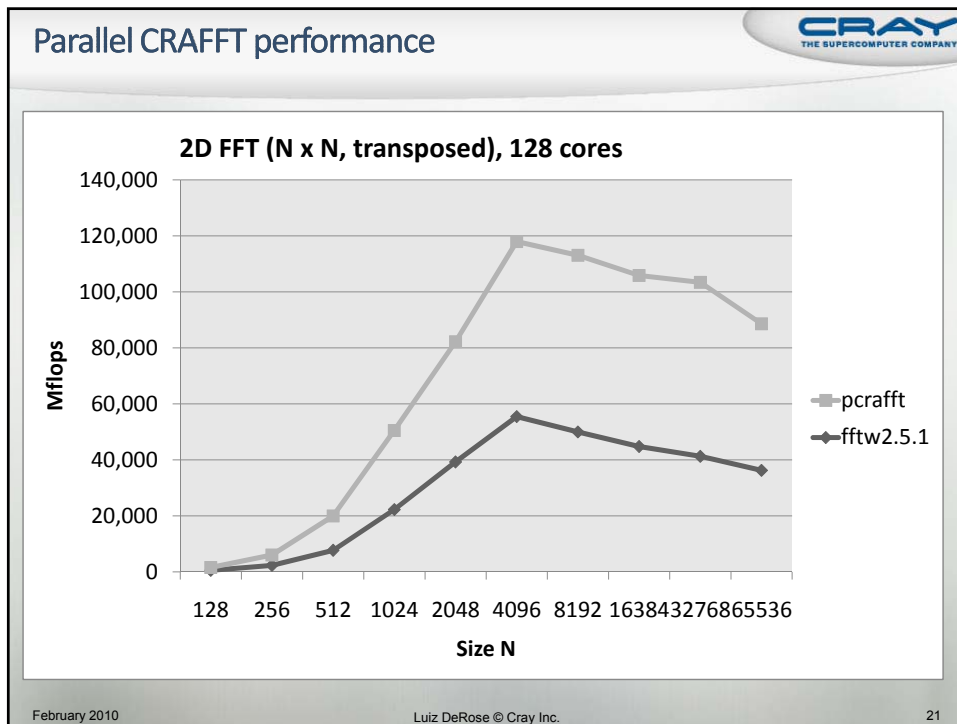
Each routine above has man page. Also see 3d equivalent :

```
man crafft_pz2z3d
```

February 2010

Luiz DeRose © Cray Inc.

20



Iterative Refinement Toolkit

CRAY
THE SUPERCOMPUTER COMPANY

- Solves linear systems in single precision
- Obtaining solutions accurate to double precision
 - For well conditioned problems
- Serial and Parallel versions of LU, Cholesky, and QR
- 2 usage methods
 - **IRT Benchmark routines**
 - Uses IRT 'under-the-covers' without changing your code
 - Simply set an environment variable
 - Useful when you cannot alter source code
 - **Advanced IRT API**
 - If greater control of the iterative refinement process is required
 - Allows
 - » condition number estimation
 - » error bounds return
 - » minimization of either forward or backward error
 - » 'fall back' to full precision if the condition number is too high
 - » max number of iterations can be altered by users

February 2010

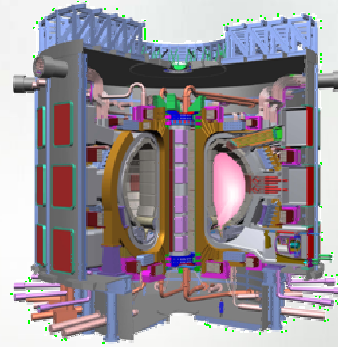
Luiz DeRose © Cray Inc.

22

Example: AORSA Fusion Energy

CRAY
THE SUPERCOMPUTER COMPANY

- “High Power Electromagnetic Wave Heating in the ITER Burning Plasma”
- rf heating in tokamak
- Maxwell-Boltzmann Eqns
- FFT
- Dense linear system
- Calc Quasi-linear op



ITER-FEAT

Courtesy
Richard Barrett
**OAK
RIDGE**
National Laboratory



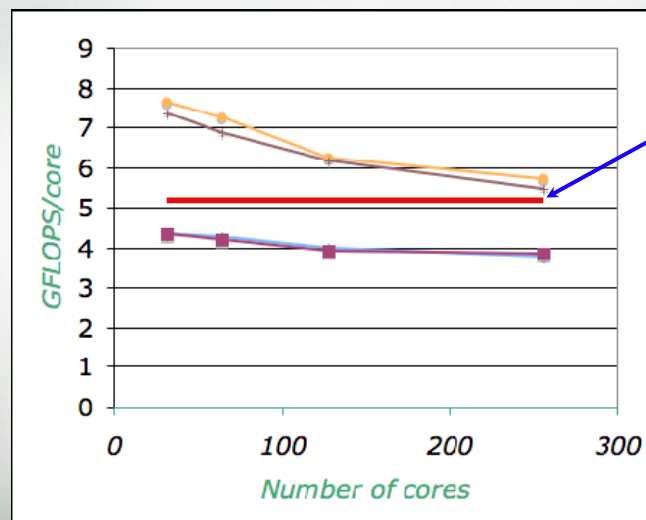
February 2010

Luiz DeRose © Cray Inc.

23

Example: AORSA Fusion – solver performance on 128x128 grid

CRAY
THE SUPERCOMPUTER COMPANY



Theoretical
Peak

OAK
RIDGE
National Laboratory

February 2010

Luiz DeRose © Cray Inc.

24

IRT usage



Decide if you want to use advanced API or benchmark API

benchmark API :

`setenv IRT_USE_SOLVERS 1`

Advanced API :

1. locate the factor and solve in your code (LAPACK or ScaLAPACK)
2. Replace factor and solve with a call to IRT routine
 - e.g. `dgesv` -> `irt_lu_real_serial`
 - e.g. `pzgesv` -> `irt_lu_complex_parallel`
 - e.g. `pzposv` -> `irt_po_complex_parallel`
3. Set advanced arguments
 - Forward error convergence for most accurate solution
 - Condition number estimate
 - "fall-back" to full precision if condition number too high

February 2010

Luiz DeRose © Cray Inc.

25

Upcoming Releases

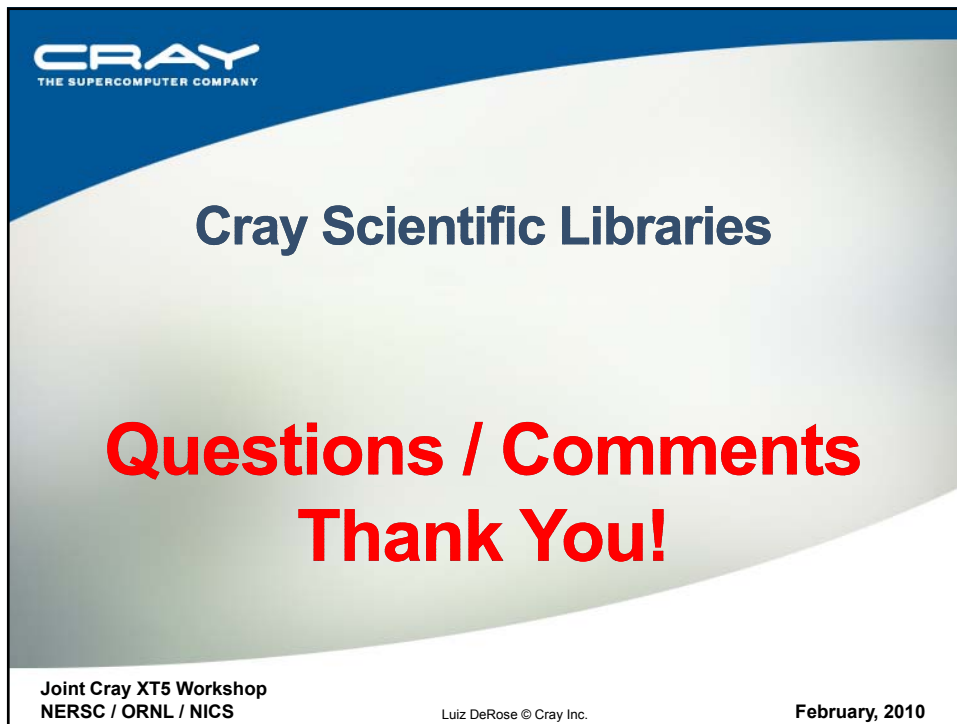


- LibSci 10.4.2 February 18th 2010
 - OpenMP-aware LibSci
 - Allows calling of BLAS inside or outside parallel region
 - Single library supported
 - No multi-thread library and single thread library (`-lsci` and `-lsci_mp`)
 - Performance not compromised
- Trilinos update February 18th 2010
 - CASK optimizations for multiple vectors (as shown earlier)

February 2010

Luiz DeRose © Cray Inc.

26



The slide features a blue header with the Cray logo and a light gray background with a blue wave-like border. The text is centered and uses a mix of blue and red colors for emphasis.

CRAY
THE SUPERCOMPUTER COMPANY

Cray Scientific Libraries

Questions / Comments
Thank You!

Joint Cray XT5 Workshop
NERSC / ORNL / NICS

Luiz DeRose © Cray Inc.

February, 2010